KNN+Confusion Matrix

Prepared For:

Mr. Henry Chang
Machine Learning and Business Intelligence CS550
Summer 2021
Northwestern Polytechnic University

Prepared By:

Ms. Nagalla, Santhi Sree ID:19568

Table Of Contents

- What is Confusion Matrix?
- Implementation
- Description of Each Cell
- Basic Rules To Follow
- Calculate:
  - Confusion matrix
  - Accuracy
  - Precision
  - Recall
  - F1 Score
- Conclusion

What is Confusion Matrix?

- Confusion matrix is one such important tool which helps us evaluate our model’s performance. As the name suggests it is a matrix of size n x n .where ’n’ is the number of class labels in our problem.
- Let’s take a look at confusion matrix structure. Here,I am showing the python standard matrix notation for two class classification.

Implementation

- In Python implementation of confusion matrix, rows show actual values and columns indicate predicted values. Given below is the description of each cell.

		Predicted values		
		Positive	Negative	Totals
Actual Values	Positive	TP	FN	P = (TP + FN) = Actual Total Positives
	Negative	FP	TN	N = (FP + TN) = Actual Total Negatives
	Totals	Predicted Total Positives	Predicted Total Negatives	

Python's Representation of Confusion Matrix

Description of Each Cell

TP (True Positives):

Actual positives in the data, which have been correctly predicted as positive by our model. Hence True Positive.

TN (True Negatives):

Actual Negatives in the data, which have been correctly predicted as negative by our model. Hence True negative.

FP (False Positives):

Actual Negatives in data, but our model has predicted them as Positive. Hence False Positive.

FN (False Negatives):

Actual Positives in data, but our model has predicted them as Negative. Hence False Negative.

Basic Rules To Follow

- Rows are always actual and columns are predictions. This is the base rule.
- Once we set that base rule, first cell is TP. Diagonal to it is TN.
- Once we set TP and TN, come to the row where we have TP. Next to it comes the opposite of T and opposite of P. So, opposite of T is F and opposite of P is N. So, we put FN next to it.
- Similarly in the second row, where we have TN, we put opposite of it in the next cell. So, we put FP.

Confusion matrix

- Now, if we calculate confusion matrix of our model using above formula.
- If the objective is to determine the "+" class for below 2 Scenarios.

K=3		Predicted Assessment	
Correct Assessment		+	-
	+	12	1
	-	1	11

TP + ==> + 12	FP ! +(-)==> + 1
FN + ==> !+(-) 1	TN !+(-) ==> !+(-) 11

K=5		Predicted Assessment	
Correct Assessment		+	-
	+	3	7
	-	7	8

TP + ==> + 3	FP ! +(-)==> + 7
FN + ==> !+(-) 7	TN !+(-) ==> !+(-) 8

Accuracy

In ML, We have so many metrics. Out of which most known and used one is Accuracy.

Accuracy:

$$Accuracy = \frac{TP + TN}{P + N} = \frac{TP + TN}{TP + FN + TN + FP}$$

Accuracy formula:

Accuracy tells the percentage of correctly predicted values out of all the data points. Often times, it may not be the accurate metric for our model performance. Specifically, when our data set is imbalanced. Let's assume I have a data set with 100 points, in which 95 are positive and 5 are negative.

Accuracy

- Now, if we calculate accuracy of our model using above formula.

For Model K=3 ⇒ Accuracy = 12+11/12+1+1+11 ⇒ 0.92

For Model K=5 ⇒ Accuracy = 3+8/3+7+7+8 ⇒ 0.44

Assuming that our test data has more credit approval cases and less credit denial cases, which generally will be the case. Our model needs to identify negative more accurately than Positive.

we understood that accuracy is not always the best metric and different models will have different metrics based on business case. Let’s see some of these metrics and how to remember them.

TPR (True Positive Rate) or Recall

TPR (True Positive Rate) or Recall:

$$TPR = \frac{TP}{TP + FN}$$

TPR formula -

It tells us, out of all the Credit approval cases , how many have been truly identified as positive by our model.

TPR is TP divided by the values of the row in which TP is present.

Ex - Our Model k= 3 ⇒ Recall ⇒ 12/12+1 ⇒ 0.92

Our Model k= 5 ⇒ Recall ⇒ 3/3+7 ⇒ 0.3

Precision

Precision:

$$Precision = \frac{TP}{TP + FP}$$

TNR formula -

It tells use, out of all the points which have been identified as positive by our model, how many are actually true.

Ex - Our Model k= 3 ⇒ Precision⇒ 12/12+1 ⇒ 0.92

Our Model k= 5 ⇒ Precision ⇒ 3/3+7 ⇒ 0.3

F1 Score

- F1 score is a simple way to compare two classifiers.
- The F1 score is the harmonic mean of precision and recall
- Whereas the regular mean treats all values equally, the harmonic mean gives much more weight to low values.
- As a result, the classifier will only get a high F1 score if both recall and precision are high.

$$F_1 = \frac{2}{\frac{1}{\text{precision}} + \frac{1}{\text{recall}}} = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} = \frac{TP}{TP + \frac{FN+FP}{2}}$$

Ex - For Model K=3 ⇒ F1 = 12/12 + 1⇒ 12/13 ⇒ 0.92

For Model K=5 ⇒ F1 = ⇒ 3/3 + 7 ⇒ 3/10 ⇒ 0.3

Conclusion

- F1 Score is the weighted average of Precision and Recall. Therefore, this score takes both false positives and false negatives into account. Intuitively it is not as easy to understand as accuracy, but F1 is usually more useful than accuracy, especially if you have an uneven class distribution. Accuracy works best if false positives and false negatives have similar cost. If the cost of false positives and false negatives are very different, it's better to look at both Precision and Recall.
- In our case, F1 score is 0.92 for Model K=3 and 0.3 for Model K=5.
- So Model K=3 is best.

F1 Score = 2*(Recall * Precision) / (Recall + Precision)

K=	TP	FN	FP	TN	Precision	Accuracy	Recall	F1 score
3	12	1	1	11	0.92	0.92	0.92	0.92
5	3	7	7	8	0.3	0.44	0.3	0.3

References

- https://ai.plainenglish.io/understanding-confusion-matrix-and-applying-it-on-knn-classifier-on-iris-dataset-b57f85d05cd8
- https://npu85.npu.edu/~henry/npu/classes/data_science/algorithm/slide/Pick_an_evaluation_metric.html
- https://npu85.npu.edu/~henry/npu/classes/hands_on_ml_with_schikit_2nd/classification/slide/Precision_and_Recall.html
- Google Slides URL -

  https://docs.google.com/presentation/d/1AEtPa-TIvQlHOnyjRQNS-Mcdw04Rx_2PQf7gEaLjOp8/edit?usp=sharing

- GitHub URL -

  https://github.com/santhinagalla/Machine-Learning/tree/main/Supervised%20Learning/KNN%2BConfusionMatrix